

In the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

1 1. (Currently Amended) A processor having a changeable
2 architected state, comprising:
3 instruction memory for storing instructions;
4 an instruction pipeline, wherein an instruction which passes
5 entirely through the pipeline alters the architected state and
6 wherein the pipeline comprises circuitry for fetching instructions
7 from the instruction memory into the pipeline;
8 ~~circuitry~~ an annul word memory for storing an annul code
9 having a plurality of annul bits, ~~an~~ each annul bit corresponding
10 ~~to each~~ having a one-to-one correspondence to one instruction of a
11 group of instructions in the pipeline; and
12 circuitry for preventing one or more selected instructions in
13 the group from altering the architected state in response to the
14 corresponding annul bit of the annul code.

Claims 2 to 4. (Canceled)

1 5. (Currently Amended) The processor of claim 3 1:
2 wherein the instruction pipeline further comprises a plurality
3 of execution units;
4 wherein the plurality of execution units are operable such
5 that in a given clock cycle an integer number N of the plurality of
6 execution units are scheduled to execute; and
7 wherein the circuitry for preventing one or more selected
8 instructions in the group from altering the architected state
9 comprises circuitry for coupling annul bits to respective ones of
10 the plurality of execution units; and

11 wherein the circuitry for coupling the annul bits to
12 respective ones of the plurality of execution units comprises
13 circuitry for coupling only the integer number N of the annul bits
14 to the plurality of execution units which are scheduled to execute
15 in the given clock cycle whereby on an immediately following clock
16 cycle annul bits beginning at an N+1 annul bit are coupled to
17 execution units scheduled to execute of that following cycle.

Claim 6. (Canceled)

1 7. (Currently Amended) The processor of claim 5:
2 wherein the group of instructions corresponding to the annul
3 code comprise instructions corresponding to a software loop
4 scheduled to execute for an integer M number of iterations; and
5 wherein during a given iteration the circuitry for preventing
6 prevents one or more of the group of instructions corresponding to
7 the annul bits of the annul code from altering the architected
8 state in response to the annul bits of the annul code and the annul
9 code based on a relationship of the given iteration to the integer
10 M number of iterations preventing differing instructions from
11 altering the architected state during different iterations.

Claim 8 and 9. (Canceled)

1 10. (Currently Amended) The processor of claim 1 wherein the
2 annul code is generated in response to one or more constant
3 generating instructions and loaded into the annul word memory.

1 11. (Currently Amended) The processor of claim 1 wherein the
2 annul code is loaded into the annul word memory from a memory.

1 12. (Currently Amended) The processor of claim 1 wherein the
2 annul code is an immediate value in an immediate operand
3 instruction passing through the pipeline loaded into the annul word
4 memory in response to execution of the immediate operand
5 instruction.

Claims 13 to 15. (Canceled)

1 16. (Currently Amended) The processor of claim 1:
2 wherein the annul code is loaded into the annul word memory
3 from a selected location of memory in response to an instruction
4 having a condition predicate;
5 wherein the annul code comprises a first annul code stored in
6 a first location in the memory loaded into the annul word memory in
7 response to the condition predicate being satisfied; and
8 wherein the annul code comprises a second annul code stored in
9 a second location in the memory loaded into the annul word memory
10 in response to the condition predicate not being satisfied.

1 17. (Currently Amended) The processor of claim ~~16~~ 1:
2 and further comprising a first data register and a second data
3 register;
4 wherein the annul code is loaded into the annul word memory
5 from a selected one of the first data register and the second data
6 register in response to an instruction having a condition
7 predicate;
8 wherein the annul code comprises a first annul code is stored
9 in the first data register loaded into the annul word memory in
10 response to the condition predicate being satisfied; and
11 wherein the annul code comprises a second annul code is stored
12 in the second data register loaded into the annul word memory in
13 response to the condition predicate not being satisfied.

1 18. (Currently Amended) The processor of claim ~~16~~ 1:
2 and further comprising a data register;
3 wherein the annul code is loaded into the annul word memory
4 from a selected half of the data register in response to an
5 instruction having a condition predicate;
6 wherein the annul code comprises a first annul code ~~is~~ stored
7 in a first one-half of the data register loaded into the annul word
8 memory in response to the condition predicate being satisfied; and
9 wherein the annul code comprises a second annul code ~~is~~ stored
10 in a second one-half of the data register different from the first
11 one-half loaded into the annul word memory in response to the
12 condition predicate not being satisfied.

Claim 19. (Canceled)

1 20. (Currently Amended) The processor of claim 1:
2 and further comprising a register;
3 wherein the register stores the annul code which comprises a
4 set of annul bits having a first logical value and a set of annul
5 bits having a second logical value;
6 wherein the annul code is loaded into the annul word memory
7 from the register in response to an instruction having a condition
8 predicate;
9 wherein the circuitry for preventing prevents instructions
10 corresponding to annul bits having a first logical value from
11 altering the architected state in response to the condition
12 predicate being satisfied; and
13 wherein the circuitry for preventing prevents instructions
14 corresponding to annul bits having a second logical value opposite
15 the first logical state from altering the architected state in
16 response to the condition predicate not being satisfied.

1 21. (Currently Amended) The processor of claim 1 and further
2 comprising circuitry for storing a portion of the annul code into
3 the annul word memory in response to receipt of an interrupt.

1 22. (Currently Amended) A method of data processing
2 comprising the steps of:

3 identifying at compile time prior to execution a group of
4 instructions including a tree of a plurality of conditional branch
5 instructions;

6 at compile time prior to execution for each conditional branch
7 instruction within the group of instructions

8 forming a first annul code having an annul bit
9 corresponding to each instruction following the conditional
10 branch instruction, the annul bit having a first logical state
11 for instructions following the detected conditional branch
12 instruction executed if a condition of the conditional branch
13 instruction is satisfied and a second logical state opposite
14 to the first logical state for instructions following the
15 detected conditional branch instruction executed if the
16 condition of the conditional branch instruction is not
17 satisfied,

18 forming a second annul code having an annul bit
19 corresponding to each instruction following the conditional
20 branch instruction, the annul bit having the second logical
21 state for instructions following the detected conditional
22 branch instruction executed if a condition of the conditional
23 branch instruction is satisfied and the first logical state
24 for instructions following the detected conditional branch
25 instruction executed if the condition of the conditional
26 branch instruction is not satisfied;
27 upon execution of the group of instructions

28 detecting each conditional branch instruction,
29 evaluating the condition of the conditional branch
30 instruction,
31 loading the corresponding first annul code if the
32 condition of the conditional branch instruction is satisfied,
33 loading the corresponding second annul code if the
34 condition of the conditional branch instruction is not
35 satisfied,
36 executing each instruction following the conditional
37 branch instruction if the corresponding annul bit of the
38 corresponding annul code has the first state, and
39 not executing each instruction following the conditional
40 branch instruction if the corresponding annul bit of the
41 corresponding annul code has the second state.